SGRel-DiT: Scene-Graph-Aware Relation Analysis in Diffusion Transformers

Author Name
Affiliation
email@example.com

December 15, 2025

Abstract

We present SGRel-DiT, a scene-graph-aware diffusion-transformer framework for relation analysis in generative diffusion models. This report documents the motivation, formal theory, architecture, training objectives, and experimental protocol used in the project. We provide mathematical derivations for the diffusion-based generative backbone, the scene-graph encoding and fusion into transformer layers, and relation-level supervision and evaluation.

Keywords: diffusion models, transformers, scene graphs, relation analysis, interpretability, LoRA

1 Introduction

High-fidelity image synthesis using diffusion models has advanced rapidly. However, controlling and analyzing relations between objects in generated scenes remains challenging. We propose SGRel-DiT, a method that augments diffusion transformers with scene-graph-aware modules to explicitly encode objects and relations and to analyze relation representations learned by diffusion attention mechanisms.

Contributions:

- A formalization of scene-graph-conditioned diffusion transformer architectures.
- Losses and evaluation protocols for relation analysis and classification.
- An empirical study of which diffusion-transformer layers best encode relations using ConceptAttention saliency, including layer-group datasets (early/middle/late) and classifier benchmarks.

2 Related Work

We build on work in diffusion models [3, 5], concept-level attention in diffusion transformers [2], LoRA-style parameter efficient adaptation [4], and scene-graph reasoning literature. See also SGDiff and related project materials [1].

3 Background and Notation

We denote observed images by $\mathbf{x}_0 \in \mathbb{R}^{H\times W\times 3}$. A diffusion model defines a forward noising process and a learned reverse denoising process. The forward (Gaussian) process for timesteps $t = 1 \dots T$ is:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1}, \beta_t\mathbf{I}),$$

with schedule $\{\beta_t\}_{t=1}^T$. The marginal at time t satisfies:

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}),$$

where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^t \alpha_s$.

The denoising model is parameterized as $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$, often implemented by predicting the noise $\epsilon_\theta(\mathbf{x}_t, t)$; the simplified training objective is:

$$\mathcal{L}(\theta) = \mathbb{E}_{\mathbf{x}_0,\epsilon,t}\left[\|\epsilon - \epsilon_\theta(\mathbf{x}_t, t)\|_2^2\right].$$

3.1 Transformers and Attention

We use standard multi-head self-attention. Given queries Q, keys K, and values V, the attention output is:

$$\text{Attn}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V.$$

3.2 Scene Graphs

A scene graph is a directed graph $G = (V, E)$ where nodes $v \in V$ represent objects with attributes and edges $e_{ij} \in E$ represent relations (e.g., "on", "next to"). We assume a set of object bounding boxes and labels $\{(b_i, y_i)\}_{i=1}^n$ and relation labels r_{ij} when available.

Node embeddings are written $\mathbf{h}_i \in \mathbb{R}^d$ and edge embeddings $\mathbf{e}_{ij} \in \mathbb{R}^{d_e}$. We encode scene graphs into a transformer-friendly sequence by concatenating object tokens, relation tokens, and global tokens with learned position encodings.

4 Method: SGRel-DiT

We describe how a scene graph conditions the diffusion transformer and how we analyze relation representations.

4.1 Graph-conditioned diffusion transformer

We augment the diffusion transformer backbone by injecting scene-graph encodings via cross-attention at multiple layers. For an intermediate image-token representation $X \in \mathbb{R}^{N\times d}$ and graph token representations $G \in \mathbb{R}^{M\times d}$, we introduce cross-attention blocks:

$$X' = \text{MHA}\left(Q = \phi_q(X), K = \phi_k(G), V = \phi_v(G)\right) + X,$$

where $\phi_\cdot$ are linear projections. The cross-attention allows object and relation tokens to modulate image generation and representation at each time step.

4.2 Graph encoder: triple convolution network

To embed a discrete scene graph into continuous tokens we use a message-passing encoder aligned with prior SG reasoning stacks. Given object vectors $\{\mathbf{h}_i\}_{i=1}^n$ and predicate (edge) vectors $\{\mathbf{p}_{ij}\}$, and a list of directed edges $E = \{(i, j)\}$, a single triple-convolution layer updates subject/predicate/object jointly:

$$\begin{aligned}
\mathbf{t}_{ij} &= [\mathbf{h}_i;\, \mathbf{p}_{ij};\, \mathbf{h}_j], \\
[\Delta\mathbf{h}_i^{(ij)};\, \mathbf{p}'_{ij};\, \Delta\mathbf{h}_j^{(ij)}] &= f_{\text{triple}}(\mathbf{t}_{ij}), \\
\tilde{\mathbf{h}}_i &= \text{Pool}_{j:(i,j)\in E}(\Delta\mathbf{h}_i^{(ij)}), \quad \tilde{\mathbf{h}}_j = \text{Pool}_{i:(i,j)\in E}(\Delta\mathbf{h}_j^{(ij)}), \\
\mathbf{h}'_i &= f_{\text{obj}}(\tilde{\mathbf{h}}_i).
\end{aligned}$$

Predicate embeddings $\mathbf{p}'_{ij}$ are updated by f_{triple}. Here f_{triple} and f_{obj} are MLPs and Pool is sum/average pooling across incident edges. Stacking L layers yields final object/predicate embeddings that are serialized into graph tokens G and injected via cross-attention.

4.3 ConceptAttention side-stream and saliency readout

Our relation-analysis pipeline uses ConceptAttention [2] to extract per-layer concept saliency without changing the generative computation. For a multi-modal diffusion transformer block ℓ, let X_ℓ denote image-token states and let C_ℓ denote an auxiliary set of concept-token states derived from a small list of prompt concepts (e.g., subject, predicate, object). Concept tokens are encoded with the same text encoder used for prompts.

ConceptAttention performs a one-directional update: concept queries attend over image keys/values (and optionally concept keys/values) while the image stream remains unchanged by concept tokens. Writing projections in block ℓ as $W_Q^{(\ell)}, W_K^{(\ell)}, W_V^{(\ell)}$ (shared with the text projections for that block), we compute

$$\begin{aligned}
Q_C &= W_Q^{(\ell)}\ \mathrm{LN}(C_\ell),\\
K_C &= W_K^{(\ell)}\ \mathrm{LN}(C_\ell), \quad V_C = W_V^{(\ell)}\ \mathrm{LN}(C_\ell),\\
K &= [K_X \,|\, K_C], \quad V = [V_X \,|\, V_C],\\
O_C &= \mathrm{Attn}(Q_C, K, V), \quad C_{\ell+1} = C_\ell + g(O_C),
\end{aligned}$$

where K_X, V_X are the image-stream keys/values for the same block and $g(\cdot)$ denotes the block output projection and feed-forward update used for the text stream.

For interpretation, ConceptAttention defines an output-space saliency for concept c at image token i as a dot product between image and concept outputs:

$$S_{\ell,c}(i) = \langle o_{\ell,i}^{\text{img}},\, o_{\ell,c}^{\text{concept}} \rangle.$$

Reshaping $S_{\ell,c}$ to the patch grid yields a per-layer saliency heatmap. We aggregate saliency across diffusion timesteps (average) and across layer groups (early/middle/late) to obtain features for relation classification.

4.4 Relation-aware objectives

To analyze relation representations we include auxiliary losses. For relation classification (given node pair (i, j)):

$$\mathcal{L}_{rel} = -\sum_{(i,j)\in P} \log p_\phi(r_{ij} | \mathbf{z}_{ij}),$$

where $\mathbf{z}_{ij}$ is a learned pooling of token activations relevant to (i, j) (e.g., cross-attention weights or concatenated object features), and p_ϕ is a softmax classifier.

We also optionally supervise attention maps using a relational saliency loss when human saliency masks or synthetic ground-truth attentions are available.

4.5 Parameter-efficient adaptation (LoRA)

For experiments with fine-tuning we support LoRA-style low-rank adapters inserted into attention projections (see [4]). If a projection $W \in \mathbb{R}^{d\times d}$ is adapted, LoRA parameterizes the update as $W + BA$ with $B \in \mathbb{R}^{d\times r}, A \in \mathbb{R}^{r\times d}$ and rank $r \ll d$.

5 Mathematical Details

5.1 Relation representation pooling

Given per-token features $\{\mathbf{t}_k\}_{k=1}^{N}$, we form pairwise pooled features for nodes i, j by

$$\mathbf{z}_{ij} = \text{Pool}\left([\mathbf{h}_i, \mathbf{h}_j, \mathbf{h}_i - \mathbf{h}_j, \mathbf{h}_i \odot \mathbf{h}_j]\right),$$

where Pool can be concatenation followed by an MLP; $\odot$ denotes elementwise product. A relation classifier is then $p_\phi(r|\mathbf{z}_{ij}) = \text{softmax}(W_r \mathbf{z}_{ij} + b_r)$.

5.2 Joint training objective

The total training objective is a weighted sum:

$$\mathcal{L}_{total} = \mathcal{L}_{diff} + \lambda_{rel}\mathcal{L}_{rel} + \lambda_{attn}\mathcal{L}_{attn} + \lambda_{reg}\mathcal{L}_{reg},$$

where $\mathcal{L}_{diff}$ is the DDPM noise-prediction loss, $\mathcal{L}_{rel}$ is relation classification loss, $\mathcal{L}_{attn}$ is optional attention supervision, and $\mathcal{L}_{reg}$ regularizes adapter weights when used.

6 Experiments

This section describes datasets, splits, metrics, and implementation details.

6.1 Datasets and splits

Stage A (prompt + concept construction). We normalize Visual Genome relationship triples into a 24-class predicate set spanning geometric and semantic relations (9 geometric: left/right/above/below/in-front/behind/in/on/near; 15 semantic: holding, carrying, wearing, riding, looking at, touching, using, pushing, pulling, sitting on, standing on, hanging from, eating, drinking, playing with). Each example emits a JSONL record containing a canonical triple, a prompt template instantiation, and a concept list (typically [subject, predicate, object]). This stage is implemented by the project scripts (see repository Stage A guide).

Stage B (ConceptAttention tracing). We run a diffusion-transformer backbone (Flux-small / FLUX.1-schnell in our prototype) with a ConceptAttention tracer that monkey-patches transformer blocks to log per-block image/concept outputs and compute saliency maps. We use a small number of denoising steps for throughput (e.g., 4 steps) and fixed seeds for reproducibility.

Saliency map datasets. Using the tracer outputs, we generate three supervised datasets corresponding to early/middle/late transformer layer groups:

- Early layers: $\{0, 1, 2, 3, 4, 5, 6\}$,
- Middle layers: $\{7, 8, 9, 10, 11, 12\}$,
- Late layers: $\{13, 14, 15, 16, 17, 18\}$.

Each dataset contains 24 predicate classes with a target of 2000 samples/class (48,000 samples/dataset). Classes with fewer source examples are oversampled with shuffling to equalize class counts.

Each sample stores: (i) output-space ConceptAttention saliency maps, (ii) cross-attention maps (baseline), (iii) prompt and concept list, (iv) predicate/class id, (v) layer configuration metadata.

Data splits and balancing. The dataset generator targets balanced classes (2000 samples per predicate). For predicates with fewer unique Visual Genome examples (e.g., "right of" or "pushing"), the pipeline oversamples source examples with shuffling to reach the target count. For classifier training we use fixed train/val/test splits produced by the experiment scripts (70/15/15 splits are used in the longer sweeps described below).

6.2 Metrics

Relation classification. We report top-1 accuracy and macro-F1 across the 24 predicates, with confusion matrices stratified by geometric vs semantic subsets.

Graph-conditioning sensitivity and alignment. For graph-conditioned generation experiments we compare outputs under a positive graph g^+ (matching the prompt) and a negative graph g^- (mismatched predicate/object swaps). We report pixel MSE and LPIPS between generated images from g^+ and g^- as a sensitivity proxy, and CLIP-text alignment margins as a semantic proxy (see Section 7.3).

6.3 Implementation details

Dataset generation. We generate saliency datasets with a multi-GPU pipeline (data-parallel over samples). Images are rendered at a fixed resolution (512×512 for the saliency generator; 32×32 patch grid) with a small number of denoising steps for throughput; saliency is averaged over diffusion timesteps and optionally over layer groups.

Relation classifier. We train a WideResNet-28-8 classifier on saliency maps, cross-attention maps, and their simple fusion variants. Unless otherwise stated, we use AdamW with learning rate 10^{-3} and weight decay 10^{-4}, batch size 128, 5 epochs, and dropout 0.3.

Layer-comparison sweeps. To compare early/middle/late layer groups at higher accuracy, the repository also includes longer sweeps using WideResNet-28-10 with dropout 0.3, SGD (lr 0.01, momentum 0.9, wd $5 \cdot 10^{-4}$), batch size 64 (and an additional "latest" sweep at 100 epochs, batch 128). These sweeps evaluate multiple fusion operators (concat/triple/diff/late fusion and baselines).

Graph-conditioned diffusion + LoRA. For graph conditioning experiments we insert LoRA adapters into selected transformer blocks and evaluate both sensitivity (MSE/LPIPS) and text alignment (CLIP). Example configurations include LoRA rank $r = 16$ with adapters applied to mid blocks (e.g., blocks 7–13), with low-step sampling for evaluation.

6.4 Figures and tables

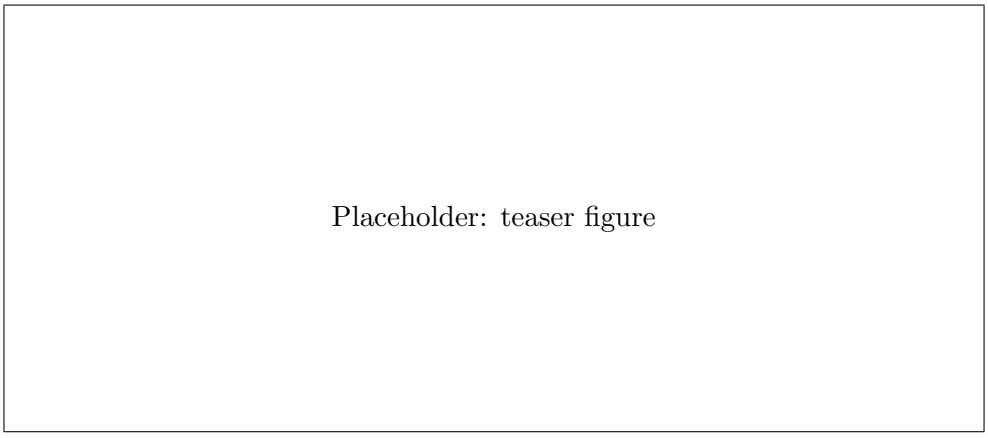

Figure 1: Teaser: example conditioned generations and relation overlays.

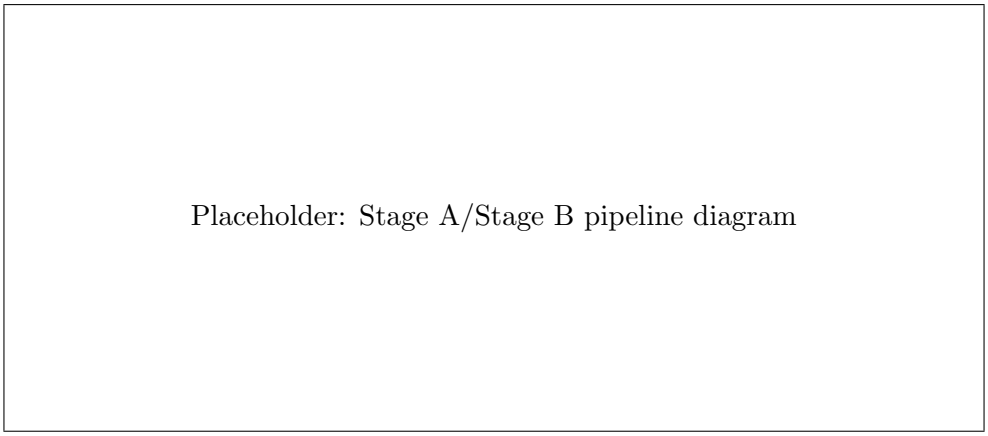

Figure 2: Pipeline overview: (Stage A) Visual Genome triples → prompts+concepts; (Stage B) ConceptAttention tracing → saliency maps; (Stage C) relation classifier training/evaluation.

Table 1: Summary of experimental settings and hyperparameters.

Setting	Value	Notes
Backbone (saliency)	Flux-small (FLUX.1-schnell)	Prototype ConceptAttention tracer
Steps (saliency)	4	Throughput-oriented tracing
Resolution (saliency)	512×512	32×32 patch grid
Layer groups	early/middle/late	Indices in text
Classes	24 predicates	Balanced via oversampling
Classifier	WideResNet-28-8	Fusion modes in text
LoRA rank (graph)	16	Example setting; mid blocks (7–13)

7 Results and Analysis

We report (i) relation classification from ConceptAttention-derived saliency maps across transformer layer groups and (ii) preliminary graph-conditioning evaluation for diffusion generation.

7.1 Saliency-based relation classification

We benchmark which representation streams best predict predicates by training a classifier on (i) ConceptAttention output-space saliency maps and (ii) cross-attention maps, and simple fusions. On the early-layer saliency dataset (24 classes, 2000 samples/class), the following validation accuracies were observed after 5 epochs: These results suggest that ConceptAttention saliency

Table 2: Early-layer predicate classification accuracy (5 epochs).

Mode	Val. accuracy
saliency	0.553
cross-attention	0.340
concat (saliency+cross)	0.437
triple (saliency+cross+product)	0.480
diff (saliency+(cross-saliency))	0.443
late fusion (two-branch)	0.593

is substantially more informative than cross-attention alone for relation prediction, and that late fusion can further improve performance.

7.2 Early vs middle vs late layer groups (longer sweeps)

Beyond the 5-epoch baseline above, the repository contains longer classifier sweeps comparing early/middle/late layer-group datasets, with multiple model families and fusion operators. Table 3 summarizes representative validation accuracies from the latest sweeps.

Table 3: Layer-group comparison (validation accuracy, latest sweeps).

Arch	Mode	Early	Middle	Late
WRN	late fusion	0.827	0.796	0.794
WRN	triple	0.799	0.800	0.794
WRN	concat	0.798	0.794	0.796
tiny CNN	triple	0.837	0.798	0.804
MLP	best of (diff/concat/triple)	0.749	0.705	0.744

Overall, convolutional models substantially outperform MLP baselines, and fused representations (concat/triple/late) consistently outperform single-stream saliency-only or cross-attention-only inputs. Middle layers are competitive with late layers for several fusion settings, supporting the hypothesis that relational information emerges strongly in intermediate transformer blocks.

7.3 Graph-conditioned diffusion evaluation

For graph-conditioned generation (Flux + graph LoRA), we evaluate sensitivity to graph changes by generating images with a positive graph g^+ and a negative graph g^- under matched seeds. In a quick 10-sample test using a 10-epoch checkpoint, we observed strong sensitivity (high MSE between g^+ and g^- images) but weaker CLIP alignment:

- MSE mean: 6459.97 (std 3323.14)
- CLIP positive margin: 1.01
- CLIP alignment accuracy: 0.60

We treat these as preliminary and report full evaluation on a larger validation set (e.g., 160 samples, multiple seeds) as future work.

Evaluation protocol (comprehensive run). The repository includes a comprehensive evaluation script that (i) generates paired samples under g^+ and g^-, (ii) reports CLIP text-alignment margins (correct vs wrong prompts), and (iii) measures perceptual and pixel distances (LPIPS, MSE) between paired generations. For stable reporting, we recommend using ≥ 160 validation samples and multiple seeds per prompt.

8 Discussion

The saliency-based classification results indicate that ConceptAttention output-space saliency maps carry substantially richer predicate information than cross-attention maps alone, particularly when combined via simple fusion operators. This supports the qualitative claim of Helbling et al. [2] that output-space similarity between image and concept residual streams provides a sharper localization signal than raw attention weights, and extends it to relation-level discrimination.

The layer-group comparison suggests that relational cues are not confined to the latest layers: middle-layer features can achieve comparable accuracy to late-layer features under strong fusion operators (e.g., triple). This is consistent with the hypothesis that intermediate transformer

blocks encode spatial and interaction structure while later blocks increasingly specialize toward appearance and semantics.

For graph-conditioned generation with LoRA adapters, preliminary evaluation shows that the model responds to graph perturbations (high MSE between g^+ and g^- generations), but semantic alignment as measured by CLIP margins is weaker. This indicates that conditioning can be effective in changing the sample, yet may not reliably enforce the intended predicate without stronger supervision, better negatives, or longer training.

Limitations. First, some reported generation metrics are based on small-scale tests (10 samples) and should be treated as diagnostic rather than definitive. Second, class balancing via oversampling improves classifier training stability but may inflate performance on underrepresented predicates without careful test-set construction. Third, the current analysis focuses on saliency maps derived from a specific backbone and tracing configuration (e.g., low-step sampling); conclusions may shift under different samplers, step counts, or backbones.

Next steps. Natural follow-ups are: (i) run the comprehensive evaluation on a larger validation set and multiple seeds, (ii) report macro-F1 and confusion matrices by geometric vs semantic predicates, (iii) add qualitative panels for representative predicates showing saliency evolution across layers, and (iv) test alternative graph encoders or stronger conditioning paths (e.g., additional cross-attention injection points).

9 Conclusion

We presented SGRel-DiT, a scene-graph-aware framework for analyzing relations in diffusion transformers. The report formalized the diffusion and attention foundations, described a graph-encoding and conditioning interface, and implemented an analysis pipeline based on ConceptAttention saliency. Empirically, saliency-derived features support strong relation classification performance and enable controlled comparisons across transformer layer groups, helping localize where relational structure is represented.

Acknowledgements

This report was prepared as part of the SGRel-DiT project. We thank the authors of the referenced open-source libraries and research papers used in this work.

References

[1] SGDiff Authors. Sgdiff: Scene-graph diffusion and related materials, 2024. See attached sgdiff.pdf or external/SGDiff folder in repository.

[2] Alec Helbling, Tuna Han Salih Meral, Ben Hoover, Pinar Yanardag, and Duen Horng Chau. Conceptattention: Diffusion transformers learn highly interpretable features. 2025. URL `https://arxiv.org/abs/2502.04320v2`. arXiv:2502.04320v2 (attached).

[3] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 2020.

[4] LoRA Authors Placeholder. Blora / lora: Low-rank adaptation for efficient fine-tuning, 2024. See attached blora.pdf in repository.

[5] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Bj"orn Ommer. High-resolution image synthesis with latent diffusion models. *arXiv preprint arXiv:2112.10752*, 2022.

A Appendix: Additional math and derivations

Pseudocode (high level). Stage A builds a JSONL of (prompt, concepts, predicate) triples from Visual Genome; Stage B runs ConceptAttention tracing during sampling to produce saliency tensors; Stage C trains a classifier on saliency maps (and optional cross-attention maps) to predict predicates.

Additional plots/tables. Add confusion matrices, per-class accuracies, and qualitative saliency panels (early vs middle vs late) here.

B Attachment references

The PDFs attached to the repository that are referenced here should be included in the submission or cited in the bibliography. For example:

- [2] – ConceptAttention: Diffusion Transformers Learn Highly Interpretable Features (attached).
- [4] – LoRA/BLORA reference (attached).
- [1] – SGDiff (attached/related code in repo).